Research Group Social Computing
TUM School of Computation, Information and Technology
Technical University of Munich

FactTUM: Fact Verification through Knowledge Graphs and LLMs

Jingyi He✉, **Mahammad Nahmadov**✉, **and Irina Ternier**✉

TUM School of Computation, Information and Technology, Technical University of Munich

✉ holly.he@tum.de
✉ mahammad.nahmadov@tum.de
✉ Irina.ternier@tum.de

March 15, 2025

Abstract —
The spread of misinformation poses a significant challenge, particularly in politics and healthcare, where false claims can shape public opinion and influence critical decisions. Automatic fact verification has become an essential NLP task to address this issue by assessing the truthfulness of claims using reliable evidence. Traditional methods rely on labor-intensive data curation and rule-based approaches, limiting scalability. To address this, we propose a knowledge graph-based fact-verification approach that enhances structured evidence retrieval and model performance. We establish a baseline for claim-only verification using prompting, achieving consistent results across multiple runs. Fine-tuning on the FactKG dataset improves stability across reasoning types. Additionally, we introduce a subgraph retrieval method to refine evidence selection. Our experiments demonstrate that while fine-tuning enhances claim-only verification, graph-based models consistently outperform language models with faster training times, highlighting their potential for scalable fact verification.

1 Introduction

With the rapid growth of digital information, the spread of misinformation has become a significant challenge. False claims in politics can influence public opinion and elections, while misinformation in healthcare can lead to dangerous medical decisions. Automatic fact verification, the task of determining the factuality of a claim based on evidence, has become a crucial task in NLP to help mitigate the spread of misinformation[1].

Early approaches to fact verification primarily relied on textual evidence from large-scale corpora while these methods often struggle with complex reasoning and fail to capture structured factual knowledge[1]. To address this limitation, knowledge graphs (KGs) have been explored as a source of structured information, enabling fact verification through logical reasoning and graph-based retrieval methods[2].

A knowledge graph consists of nodes, representing entities like people, organizations, or events, and edges, showing how these entities are connected, as shown in Figure 1. For example, the company Meyer Werft, an entity in the knowledge graph, is linked to the ship AIDA Stella with the edge 'builder,' indicating that Meyer Werft is the builder of AIDA Stella. This relationship forms a knowledge triple: Meyer Werft, builder, AIDA Stella, where the triple represents a fact about the relationship between the two entities.

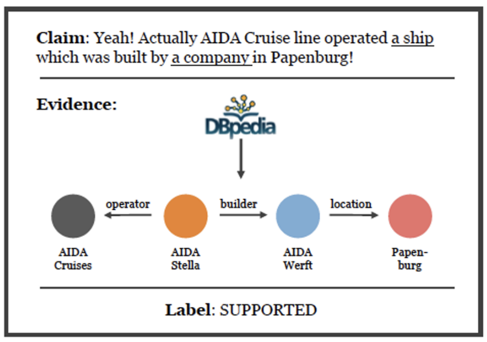

Figure 1 An example claim from FactKG[2]

In this work, we focus on fact verification using the FactKG dataset[2], which provides structured evidence in the form of knowledge graphs (KGs). The dataset includes 108,000 claims, each with different reasoning types and writing styles, supported by evidence from DBpedia[3], a large knowledge graph derived from Wikipedia. We use the following three approaches:

- **LLM Prompting**: Deploying state-of-the-art language models using a few-shot and chain-of-thought setting, without the need for additional fine-tuning. We use Llama-3.5-3B-Instruct[4] for this settings, and utilize it for subgraph retrieval.
- **Fine-tuning**: Fine-tuning pretrained models,such as Llama-3.5-3B-Instruct[4],

TinyBERT[5], on the FactKG dataset[6] to improve performance on fact verification tasks.

- **Hybrid Graph-Language Model**: Implementing modified Question-Answering in Fact-checking setting which combine LLM with Graph Neural Network (GNN).

With LLM prompting approach, we investigate how well general-purpose language models perform on the task of fact verification. Fine-tuning serves as an additional step to enhance the models' understanding of the task, leveraging the FactKG dataset. We select the QA-GNN model for its capability to process graph-based data, making it particularly suited for this problem. Our prompting approach does not require training and can function without any evidence, making it both efficient and a strong baseline.

Our main contribution is the development of models that achieve satisfactory performance (60-72% accuracy) while maintaining high efficiency – either through minimal training or no training at all. Furthermore, we propose a subgraph retrieval method for evidence retrieval, which is a key component of our approach that achieved the best performance. Our code and documentation can be found at: https://gitlab.lrz.de/00000000014AD1CD/nlp-lab-ws24_25-fact-verification.

2 Related Work

2.1 FactKG Dataset

The FactKG dataset[2] contains 108,000 English claims for fact verification, where the task is to predict the factuality of each claim. These claims are derived by applying various reasoning types to facts extracted from the DBpedia KG[3]. Each claim includes the relevant DBpedia entities and a label indicating whether the claim is factual or not.

The claims are generated using the following five reasoning types:

- **One-hop**: A one-hop claim can be verified by checking if the corresponding knowledge triple exists. utilize it for subgraph retrieval.

- **Multi-hop**: Multi-hop claims require the validation of multiple facts. For its verification, existence of a path, including multiple knowledge triples, is needed to be checked.

- **Existence**: Existence claims, which consist of head, relation or tail, relation from existing triples, can be verified by checking the presence of the entity and its relation.

- **Conjunction**:Conjunction claims consist of multiple triples representing a combination of different facts and are verified by checking the existence of all corresponding triples.

- **Negation**: Negation claims are claims in which negations (e.g., "not" or "never") are incorporated.

The dataset offers researchers a valuable resource for exploring various approaches to addressing the task of fact verification.

2.2 Quantized Low-Rank Adaptation (QLoRA)

Quantized Low-Rank Adaptation (QLoRA) is a memory-efficient fine-tuning technique designed to adapt large-scale pre-trained models with minimal computational overhead[7]. It achieves this by integrating low-rank adaptation with quantization, allowing models to be fine-tuned while keeping the majority of their parameters frozen. By maintaining a quantized representation of the base model, QLoRA significantly reduces memory usage while still enabling efficient gradient updates.

A key advantage of QLoRA is its ability to retain near full-precision performance despite operating in a lower-bit setting. This is made possible through advanced quantization techniques and optimization strategies that minimize memory consumption without sacrificing model quality. As a result, QLoRA enables fine-tuning on resource-constrained hardware while achieving performance comparable to traditional full-precision training methods. This makes it particularly suitable for our work, where hardware limitations are a significant constraint.

2.3 TinyBERT and Tiny deBERTa

TinyBERT(bert-tiny)[5] and Tiny DeBERTa (deberta-v3-small) are lightweight transformer models designed for efficient fine-tuning in resource-constrained environments. TinyBERT, a distilled version of BERT, reduces the model size and inference time while retaining much of BERT's performance, making it suitable for tasks like fact verification. Tiny DeBERTa, a smaller variant of the DeBERTa model, is optimized for performance and efficiency, particularly in general NLP tasks like claim verification.

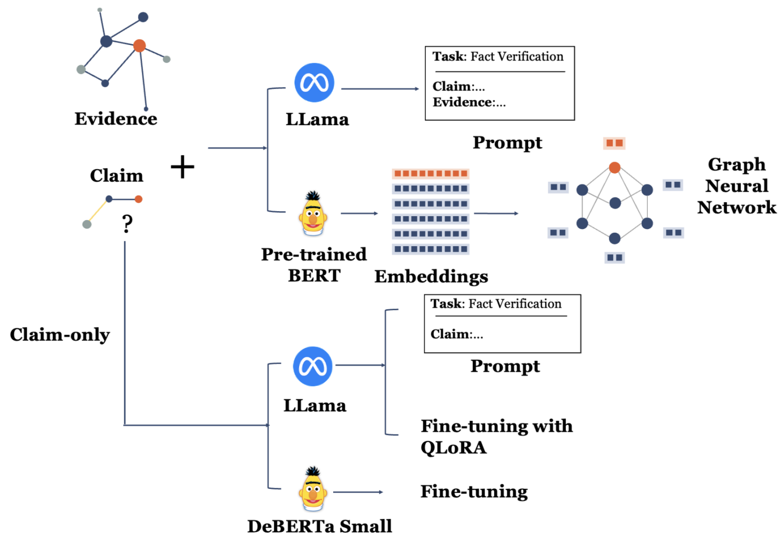

Figure 2 Our framework incorporates two fact verification approaches that differ based on the use of evidence. In the **claim-only** scenario, Llama processes the claim independently without external evidence. In the **evidence-base** scenario, relevant connections from the complete knowledge graph are first filtered using Llama. After post-processing, the extracted subgraph is integrated as supporting evidence. Finally, classifiers such as QA-GNN and Llama evaluate the claim to determine whether it is supported or refuted.

2.4 Question Answer Graph Neural Networks (QA-GNNs)

QA-GNN[8] is an end-to-end model for question answering that integrates both a language model and a graph neural network (GNN). The hybrid approach first encodes the context relevant to the question using a pretrained language model and subsequently retrieves a related subgraph. The given text is also embedded, and relevance scores are computed for each node within the subgraph. These scores serve as weights to refine the node representations. The GNN facilitates joint reasoning by iteratively updating the representations of knowledge graph (KG) entities and the question-answering context node, thereby bridging the semantic gap between them. Finally, the output from the GNN, along with the node representations and text embeddings, are concatenated before being passed through the classifier layer for the final prediction.

3 Methods

3.1 Subgraph Retrieval

To enhance computational efficiency, we explore various approaches for retrieving a relevant subgraph for each claim. Each data point in the FactKG dataset comprises a claim along with a list of entities that are present both in the claim and within the knowledge graph (KG). Given that the DBpedia subset utilized in FactKG is relatively large (1.53GB), it is essential to extract only a small, relevant subgraph as input for the models. The benchmark model[2] employs two language models to predict both the relevant edges and the depth of the extracted graph. To reduce model complexity, we aim to simplify this process by utilizing LLM for subgraph retrieval.

3.1.1 Filtering Relevant Connections

This stage leverages an LLM, specifically the Llama3-3B-Instruct model, to identify and filter relevant connections between knowledge graph (KG) entities in relation to a given claim. The overall process consists of the following key steps:

Data Preparation Entities and their corresponding relations are extracted from an enriched knowledge base (the 2015 version of DBpedia). These extracted entities and connections serve as input for the LLM.

LLM Prompting The task assigned to the LLM is explicitly formulated as filtering relevant connections to facilitate fact verification for the given claim. Given that some highly connected nodes contain over 10,000 relations, sampling is necessary to prevent prompt input overflow. A statistical analysis of the relation distribution across the KG informs the selection of 15 representative relations from the complete relation set. The filtering process consists of the following steps:

1. Structuring the prompt in a format suitable for the LLM.

2. Utilizing the LLM's inference capability to eliminate irrelevant connections based on the contextual information provided by the claim.

3. Generating a structured set of potential relations, which undergoes further post-processing in subsequent steps.

Handling Invalid LLM Responses To address potential failures in LLM responses, a retry mechanism is implemented. If the LLM output is empty or contains entities and relations not present in the given input options, the request is resubmitted, with a maximum of three attempts. Experimental results indicate that over 95% of cases yield valid responses.

3.1.2 Post-processing

First, with the filtered connections, we add 3 tail entities sampled from the original KG to form a complete triple evidence. This specific amount rises from the statistical analysis result of KG dataset, 3 is the mode value of number connecting to one specific relation.

The subgraph from filtering is further enhanced by adding direct subgraph, the difference between two types of subgraphs is illustrated in Figure 3:

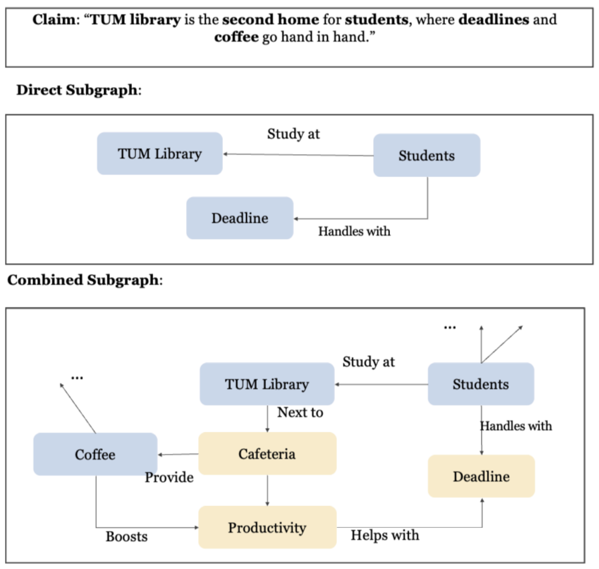

Figure 3 Examples of the different subgraphs ex- plored in this article. Boxes and bold letters represent entities, while arrows and italic letters represent rela- tions. This claim is meant for illustrative purposes and is not present in the FactKG dataset.

Reasoning type of existence only include entity and relation without a tail entity, which forms an incomplete triple, which is challenging for our filtering approach to keep. Besides, direct subgraph is the most reliable evidence for fact verification, therefore might be beneficial.

3.2 Llama Prompting

Our approach aims to evaluate the ability of state-of-the-art language models as automatic fact-checkers, establishing a useful baseline for fact verification. We design a prompt for the Llama-3.2-3B-Instruct model[4] that incorporates the claim and instructions for the model to assess the factuality of the claim.

To enhance accuracy, we employ chain-of-thought reasoning, prompting the model to provide explanations and cite sources when available. Additionally, we use a few-shot prompting strategy, including two examples of input-output pairs to guide the model's responses. Our prompt also specifies the required output format, and we implement a retry mechanism to ensure compliance with this structure and handle potential failures in LLM responses.

Due to computational constraints, we leverage the Cloudflare Workers AI REST API, which provides free access to the model with a daily usage limit. Given our test set of 9,041 claims, we complete inference in two iterations over consecutive days within the API's quota.

To address concerns about the reproducibility of language model outputs, we conduct two full test runs. While the model does not always produce identical responses for the same claim, its overall performance remains consistent, achieving 60.47% and 60.67% accuracy across the two iterations.

The Llama-3.2-3B-Instruct model is relatively lightweight and not among the latest-generation models. We anticipate that more recent state-of-the-art models could achieve significantly better performance on fact verification. Notably, our approach requires no fine-tuning, making it a strong benchmark for evaluating and contextualizing the performance of other models in this task.

3.3 Fine-Tuning Llama

We fine-tune Llama-3.2-3B-Instruct[4] as our pre-trained language models for fact verification. The models are trained using only the claims as inputs, without additional evidence, to classify whether a claim is supported or refuted. To explore optimal performance within our hardware constraints, we apply Quantized Low-Rank Adaptation (QLoRA)[7], which enables more efficient fine-tuning by reducing memory usage without sacrificing model performance. Multiple training runs are conducted with different hyperparameters, while adhering to commonly effective values—such as a learning rate of 2e-4.

Notably, our fine-tuned Llama model is the same one used in our prompting approach, allowing us to compare its behavior before and after task-specific adaptation. This provides insights into how the model's predictions shift from relying on its pretrained knowledge to leveraging patterns learned from our dataset.

To evaluate performance, we use accuracy as our primary metric across different reasoning types. This helps quantify improvements in specific reasoning capabilities resulting from fine-tuning.

3.4 Prompting with Evidence

Building upon our initial Llama prompting approach, we integrate external evidence retrieved from the subgraph retrieval module (3.1) to enhance fact verification. Our prompt incorporates both the claim and the retrieved evidence, instructing the Llama-3.2-3B-Instruct model to assess factuality based on the provided context.

We maintain a structured output format and implement a retry mechanism to enforce compliance and handle failures in LLM responses. However, incorporating evidence does not necessarily yield better results than our original prompting approach. Future work should explore refinements in integrating structured knowledge with LLMs to improve fact verification performance.

3.5 Fine-tuned tiny BERT and tiny deBERTa

This study follows a systematic approach to handle and process the FactKG[**?**] dataset for fact verification using the DBpedia. The methodology [9] includes steps for loading and analyzing the graph, computing relevant statistics, integrating graph-based features into text classification tasks, and fine-tuning models [10] for claim verification. Graph-theoretic feature extraction is performed using the NetworkX library. For each entity in the dataset, several graph-based features are calculated: the degree, which represents the number of direct connections (edges) each entity has; the clustering coefficient, which measures how well-connected the neighbors of a node are; and betweenness centrality, which indicates the importance of a node based on its position in the network. These features are then stored and added to the dataset, enriching the models with graph-based evidence. The dataset is preprocessed by adding these graph-based features to each claim. For each claim, the corresponding entities' graph features—such as degree, clustering, and betweenness centrality—are retrieved and appended to the data. Additionally, the subject, relation, and object of each claim are concatenated into a single text string and tokenized using transformer models such as TinyBERT or Tiny DeBERTa. For model training, TinyBERT and Tiny DeBERTa are used. The models are fine-tuned using the Hugging Face Trainer API, with each training example consisting of both textual input and graph-based features. Fine-tuning is conducted with a learning rate of 5e-5 for 3 epochs, using a batch size of 16.

3.6 QA-GNN architecture

To adapt the QA-GNN[8] framework to the context of fact verification, we implemented some modificaition on the model structure. Given that the potential responses are binary—either "true" or "false"—the model was adjusted to embed only the claims. Furthermore, the embedded question or claim is not directly connected to the subgraph within the model architecture.

For the language model, a pre-trained BERT model [11] was employed to facilitate the embedding process and compute relevance scores. We used frozen BERT configuration to reduce complexity for generating embeddings for both nodes and edges in the graph. Therefore, all graph embeddings can be computed beforehand. The embeddings are taken from the last hidden layer representation of the CLS token, which has a dimensionality of 768. The BERT model used for computing relevance scores and claim embeddings remain unfrozen, allowing it to dynamically learn and refine the computation of relevance scores for improved effectiveness. Relevance scores the cosine similarity between the claim embedding and the textual embeddings of the nodes.

The Graph Neural Network (GNN) component of the model includes three different flavours. Graph Attention Networks (GAT) [12] use self-attention mechanisms to assign different importance scores to neighboring nodes when updating a node's representation, while GATv2[13] improves upon GAT by modifying the attention mechanism to be more dynamic and expressive. Unlike GAT, where the attention scores are computed from static node features, GATv2 allows query-key interactions to be dynamic and learnable. TransformerConv[14] adapts the Transformer architecture[15] to graphs. Instead of using static adjacency-based aggregation like traditional GNNs, it uses a fully attention-based mechanism to learn graph structures dynamically.

Since our evidence subgraphs are quite shallow, GNN was configured with only two layers, Each layer comprises 256 features, which are mean-pooled and subsequently concatenated with the BERT embeddings before being fed into the classification layer. Dropout[16] was applied to both the GNN layers and the classification layer to reduce overfitting.

4 Results

The test results for the optimal configuration identified, along with the benchmark models, are presented in Tab. 1 and Tab. 2. In the claim-only setting, the highest-performing model was obtained from the second iteration of fine-tuning. For fact verification with evidence, the GATv2Conv model achieved the highest accuracy.

4.1 Baseline Prompting Models

Our prompting approach achieved an accuracy of 60.67%, demonstrating the capability of the Llama[4] model as a fact-checker based solely on its pretrained knowledge, without requiring any training. However, the accuracy for negation-type claims was the lowest, at 53.24%, highlighting the model's challenges in verifying such claims effectively.

To enhance performance, we incorporated evidence into the prompt, assuming that providing the model with additional context would improve accuracy. Contrary to our expectations, this resulted in a decline in overall performance, with accuracy dropping to 58.82%. However, accuracies for one-hop and negation-type claims did improve Tab. 2. This suggests that while evidence can help with certain reasoning types, further investigation is needed to understand how to best integrate evidence into our prompting approach and address the performance decline.

4.2 Finetuned Llama Models

Building on our prompting approach, we fine-tuned the Llama-3.2-3B-Instruct model on the FactKG dataset[2]. We conducted three fine-tuning experiments with different hyperparameters, achieving overall accuracies of 67.04%, 64.9%, and 63.92%.

While the first fine-tuned model achieved the highest overall accuracy (67.04%), it performed poorly on negation-type claims (53.20%) and existence-type claims (56.67%), likely due to being trained for only 500 steps, without exposure to the full training set. This pattern is similar to what we observed in the prompting approach, reflecting the pretrained model's challenges in verifying certain claims.

In contrast, the second and third fine-tuned models were trained for 3 and 2 epochs, respectively. While their overall performance was slightly lower (64.9% and 63.92%), they showed significant improvements on negation and existence-type claims. Notably, the second fine-tuned model produced consistent results across all claim types, making it the most balanced choice among models trained on claim-only input (Tab. 1).

Although we favor the second fine-tuned model for its stability and robustness, the first model's higher overall accuracy suggests the need for further investigation. Overall, fine-tuning led to significant performance gains, with the best negation-type accuracy increasing from 53.66% to 67.5%, and the overall accuracy improving from 60.47% to 64.9%.

4.3 With Evidence, GNN Achieves Higher Performance with Reduced Training Time

Given the constraints of computational resources, we sample only half of the training set for subgraph retrieval of each claim, while evaluating the results on the entire test set.

By integrating the retrieved evidence into the fact verification framework, we implement two distinct approaches: LLM Prompting and QA-GNN. The QA-GNN framework employs three variants of graph neural networks (GNNs): GATConv, GATv2Conv, and TransformerConv. As shown in Tab. 2, the QA-GNN models consistently outperform LLM Prompting in terms of accuracy on the test set. Additionally, performance differences are observed across different reasoning types. Notably, GATv2Conv and TransformerConv demonstrate significantly higher accuracy in negation-based reasoning. A possible explanation for this improvement is that GATv2Conv incorporates a more dynamic and expressive attention mechanism compared to GATConv, employing dynamic and learnable query-key interactions.

Furthermore, QA-GNN models exhibit substantially faster training times. While the GEAR model required 2–3 days of training on an RTX 3090 GPU, QA-GNN models completed training in approximately 20 minutes when using half of the training set. This

Table 1 Model Performance (Claim-only)

Model	Approach	Iteration	One-hop	Conjunction	Existence	Multi-hop	Negation	Total
Llama-3.2-3B-Instruct	Prompting	First	60.22	64.58	58.28	59.76	53.66	60.47
	Prompting	Second	61.79	64.61	57.24	59.87	53.24	60.67
	Fine Tuning	First	71.42	71.26	56.67	70.17	53.20	67.04
	Fine Tuning	**Second**	**68.50**	**63.67**	**64.60**	**61.58**	**67.50**	**64.90**
	Fine Tuning	Third	65.78	61.00	71.49	56.83	73.13	63.92

Table 2 Model Performance (With Evidence)

Model	Approach	Iteration	One-hop	Conjunction	Existence	Multi-hop	Negation	Total
Llama-3.2-3B-Instruct	Prompting	First	63.74	61.06	54.48	55.34	54.26	58.82
GATConv	Training	First	70.12	73.11	53.87	76.60	63.34	70.45
GATv2Conv	Training	First	**71.85**	**73.62**	**60.65**	**75.28**	**67.20**	**71.73**
TransformerConv	Training	First	70.47	72.60	60.32	73.40	63.37	70.50

highlights the efficiency of GNN-based fact verification in both training time and performance.

5 Conclusion

In this paper, we introduced a comprehensive framework that integrates LLM prompting, fine-tuning, and QA-GNN for fact verification. Additionally, we leveraged LLMs to filter relevant subgraphs from the original extensive knowledge base, enabling more efficient evidence retrieval. This optimized evidence mining process allows downstream models to effectively verify more complex claims.

Our experiments on the FactKG dataset demonstrated that while fine-tuning Llama improves performance in the claim-only setting, QA-GNN consistently outperforms LLM-based approaches in the evidence-supported fact verification setting. Furthermore, QA-GNN exhibits significantly faster training times, highlighting its potential to enhance fact-checking accuracy without imposing strict computational requirements.

For future work, this approach could be extended to domain-specific datasets and further improved by integrating additional sources of structured data to enhance performance.

6 Limitations and Future Work

Despite the promising integration of graph-based features into fact verification tasks, several limitations remain in the current implementation, mainly due to memory optimization challenges.

1. **Memory Constraints**
   The current approach struggles with memory limitations when processing large-scale datasets like DBpedia. Given the size of the knowledge graph and the complexity of the models involved, memory requirements exceed available resources. As a result, we were unable to generate meaningful results in some experiments. These memory constraints hinder the performance of models, especially when handling large graphs or training on resource-constrained devices.

2. **Model Optimization Needs**
   Although models such as TinyBERT and Tiny DeBERTa provide a balance between performance and efficiency, further optimization is needed to effectively integrate graph-based features without compromising model performance. The current methods require additional tuning for better scalability, particularly when dealing with larger knowledge graphs.

3. **Model Evaluation and Accuracy for graph-based method**
   While initial tests with models like BlueBERT

showed some promise with an accuracy of 0.56, it was still the least effective model compared to others referenced in related work. This suggests that further model improvements and optimizations are necessary to achieve more competitive performance levels in the task of fact verification.

To address these challenges, future work will focus on memory optimization, refining graph retrieval methods, and further tuning models to balance performance with resource usage effectively.

7 Discussion

Our filtering approach contains zero-shot prompting with large language models (LLMs) alongside necessary post-processing. The effectiveness of this two-stage process was evident. In the first stage, the LLM aids in filtering potential connections by leveraging contextual information from the claim, thereby significantly reducing the overall search space. This reduction enhances the efficiency of the subsequent post processing n stage, where direct relations are added in as supplement. The resulting filtered evidence contains around 67% of ground truth evidence.

Another noteworthy aspect is that, despite training our QA-GNN model on only half of the training set, it still achieves comparable accuracy on the test set. A possible explanation for this is that QA-GNN effectively learns patterns for extracting relevant evidence from the evidence subgraph, specifically those most closely related to the claim, thereby enhancing the fact-checking process. Future research could explore training the model on the entire dataset to further improve its accuracy and overall performance.

While the FactKG dataset is a valuable resource for fact verification, our analysis highlights key issues, particularly its imbalance. Existence (8.4%, 9,172 claims) and Negation (14.4%, 15,613 claims) claims are underrepresented, while Conjunction claims dominate (34.1%, 37,097 claims). Additionally, some entries are not true claims but rather open-ended questions, such as *"What is the name of Andy Helfer's child?"*, raising concerns about dataset consistency. A dedicated study could further assess and improve its quality.

Our results show that Llama models struggled with Existence and Negation claims, possibly due to dataset imbalance, though this remains a question. We also experimented with evidence retrieval to enhance prompting. While it did not improve overall performance, it boosted accuracy for one-hop claims, perhaps because our implementation is particularly effective for such cases.

References

[1] James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. FEVER: a large-scale dataset for fact extraction and VERification. In Marilyn Walker, Heng Ji, and Amanda Stent, editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[2] Jiho Kim, Sungjin Park, Yeonsu Kwon, Yohan Jo, James Thorne, and Edward Choi. Factkg: Fact verification via reasoning on knowledge graphs. *arXiv preprint arXiv:2305.06590*, 2023.

[3] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic web*, 6(2):167–195, 2015.

[4] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023.

[5] Naman Khurana and Vibha Prabhu. Sentiment analysis using transformer models (bert, albert, roberta, and deberta) on a smile twitter dataset. In *2024 International Conference on Innovation and Novelty in Engineering and Technology (INNOVA)*, volume I, pages 1–4, 2024.

[6] Jiho Kim, Sungjin Park, Yeonsu Kwon, Yohan Jo, James Thorne, and Edward Choi. Factkg: Fact verification via reasoning on knowledge graphs, 2023.

[7] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms, 2023.

[8] Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. Qa-gnn: Reasoning with language models and knowledge graphs for question answering. *arXiv preprint arXiv:2104.06378*, 2021.

[9] Rui Yang, Runze Wang, and Zhen-Hua Ling. Graph attention and interaction network with multi-task learning for fact verification. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7838–7842, 2021.

[10] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering*, 29(12):2724–2743, 2017.

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

[12] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks, 2018.

[13] Shaked Brody, Uri Alon, and Eran Yahav. How attentive are graph attention networks?, 2022.

[14] Yunsheng Shi, Zhengjie Huang, Shikun Feng, Hui Zhong, Wenjin Wang, and Yu Sun. Masked label prediction: Unified message passing model for semi-supervised classification, 2021.

[15] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017.

[16] Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *CoRR*, abs/1207.0580, 2012.